Inbox Payment API

Table of Contents

General description

The Inbox Payment API allows to make payments for paid services of Inbox portal, including Inbox.Games, Inbox.Mail+, etc. The API accepts payments from users by various methods - Swedbank Bank Link, SEB E-Link, SMS, PayPal, Visa.

The API allows to charge flexible fees but has fixed nominals for SMS (see Appendix A).

The Developers may specify a method of payment and amount for each transaction. The transaction will be declined when the amount is not admissible,

Available payment methods:

Method	**Code**
Swedbank BankLink	HBL
SEB E-Link	SEBL
LattelecomBPO	SMS
PayPal	PAYPAL
Credit Card (Visa/Mastercard/Amex)	CCARD

1 How it works

Actions are divided into two parts: Requests and Notifications. Requests are initiated by a third-party application (Application) and should be serialized using one of the schemes: XML, JSON, PHP.

Requests are sent to the appropriate address:

- **http://payment.inbox.lv/1.0/xml/**
- **http://payment.inbox.lv/1.0/json/**
- **http://payment.inbox.lv/1.0/php/**

Data is serialized by appropriate means, as in the request and the response. All requests use POST method of HTTP protocol.

1.1 The Developer charges the User for a service.

The Developer sends a request to the API to create a transaction, which states:

Request:

- **action** — “transactions/create”
- **dev** — GUID of the Developer (Registered by Inbox)
- **apiKey** — GUID services (Registered by Inbox)
- **price** - method of payment and fee in LVL centimes (it is possible to use different prices for different payment methods. For example 'hbl-75, sms-95'.*)* Codes of the payment methods are stated in Section 1.
- **language** — specifies interface language of the User (options RUS, ENG, LV)

- **skin** — interface of a payment form, the visual part of the payment interface in Inbox Payment where the User will be redirected. Possible values are 'popup' (optimized for the iFrame size 538x458) and 'fullpage' - a full-sized page (default setting, when 'skin' is not defined).
- **callbackURI** — URI to be called by the System to send a notification (for example, after a successful completion of the transaction).
- **returnURI** — URI to redirect the User after a successful transaction. While empty, the User will be redirected nowhere (for compatibility).
- **cancelURI** — URI to redirect the User in case of failed transaction. While empty, the User will be redirected nowhere (for compatibility).
- **tag** — an optional parameter to pass additional information about the transaction on behalf of the Developer.

Samples:

JSON

Request:

```
{
  action: 'transactions/create',
  dev: '949bd497-5869-98a4-15e1-b4dbea70b327',
  apiKey: '',
  prices: 'hbl-75,sms-95',
  language: 'en',
  skin: 'popup'
  returnURI: 'http://example.org/game15/PaymentCommited/',
  cancelURI: 'http://example.org/game15/PaymentCanceled/',
  callbackURI: 'http://example.org/game15/PaymentCanceled/',
  tag: ''
}
```

Answer:

```
        {
          id: '9f83f1ee-1006-3af4-e592-cc78da2910fe',
          link: 'http://payment.inbox.lv/api/1.0/process/e15d0d08-c5c8-
2594-e5a7-588f83d9141b/'
        }
```

1.2 The User pays for a service on page Inbox.Payment

The User is redirected to the page Inbox Payment (or iFrame), where form of payment, conditions, and the final price are shown. The User pays for the service in accordance with the method of payment chosen.

The system carries out to validate and proceed transaction and redirects the User to the page specified by the Developer, transferring data as specified by protocol.

These data are transmitted to the Developer in a GET-request. Returned data consists of:

- transaction_uuid,
- transaction_result,
- tag.

1.3 Checking a payment status of a particular transaction

The Developer may check the status of a particular transaction using UUID of the transaction, calling the corresponding request, passing an UUID f the transaction.

- **action** — ‘transactions/check’
- **dev** — GUID of the Developer
- **id** — GUID of the transaction

Answer:

Status with the value ENUM {CREATED, IN_PROGRESS, CONFIRMED, COMPLETED, FAILED, UNKNOWN}

Sample:
JSON
Request:

```
{
  action: 'transactions/check',
  dev: '949bd497-5869-98a4-15e1-b4dbea70b327',
  id: 'd751c4b1-beaf-3ca4-a5b8-a903038b12c4',
}
```

Answer:

```
{
  status: 'COMPLETED'
}
```

1.4 Notifies of status change of a transaction to the Developer

When status of the transaction become changed, the API send notify to the Developer using callbackURI (specified on the transaction generation time). The request is sent in the following type:

```
Array
(
'transaction_uuid' => $transaction['transaction_uuid'],
'transaction_result' => 'OK',
'tag' => $transaction['tag']
)
```

Upon receiving a confirmation of a transaction status change it is required to check the status according to Section 2.3.

1.5 Controlling and monitoring of transactions

To control and monitor transactions, the API creates an account for each Developer. The administrative interface is available at http://paymenty.inbox.lv/1.0/clients

The main functions

- Monitor cash flow for each registered application
- Detailed transaction log

Appendix A

SMS code	SMS fixed nominal
01***	Ls 0.00
33	Ls 0.05
20	Ls 0.06
34	Ls 0.07
05	Ls 0.08
21	Ls 0.10
02	Ls 0.15
35	Ls 0.20
22	Ls 0.21
36	Ls 0.25
06	Ls 0.26
37	Ls 0.29
38	Ls 0.30
23	Ls 0.31
39	Ls 0.35
03	Ls 0.36
40	Ls 0.39
41	Ls 0.40
42	Ls 0.45
25	Ls 0.47
43	Ls 0.49
44	Ls 0.50
07	Ls 0.52
45	Ls 0.55
26	Ls 0.57
46	Ls 0.60
04	Ls 0.61
47	Ls 0.75
48	Ls 0.85
27	Ls 0.88
49	Ls 0.95
09	Ls 0.98
50	Ls 0.99
51	Ls 1.00
28	Ls 1.03
52	Ls 1.25
29	Ls 1.29
53	Ls 1.50
54	Ls 1.75
30	Ls 1.80
55	Ls 1.99
56	Ls 2.00
57	Ls 2.25
58	Ls 2.50
59	Ls 2.75
60	Ls 3.00

13	Ls 3.08
LVL350	Ls 3.50
LVL400	Ls 4.00
LVL450	Ls 4.50
LVL500	Ls 5.00